Report No. 7174

SRNTN-75

Unfairness Caused by Quadratic Backoff in a Packet-Radio Network

Melisse Leib

October 19, 1989

Prepared By:

BBN Systems and Technologies Corporation
10 Moulton Street
Cambridge, MA 02138

Prepared for:

DARPA/ISTO
1400 Wilson Bl.
Arlington, VA. 22209

Sponsored by:

The Defense Advanced Research Projects Agency
Under Contract No. MDA-903-83-C-0131

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency, the Army or the United States Government.

1. Introduction

This note contains the performance analysis of a prototype packet-radio pacing algorithm designed to prevent congestion by throttling traffic sources. In the pacing algorithm we analyze, each packet radio measures data-packet forwarding times on a neighbor-by-neighbor basis, and scales them to compute a lower bound on packet inter-transmission times, known as a pacing delay[1]. The scale factor is chosen to prevent packet radios from sending data packets to their neighbors faster than they can forward them. In summary, the pacing algorithm applies flow control to prevent congestion.

The prototype algorithm differentiates between packet transmissions at the head of a route and packet transmissions along a route. The pacing delays for the former case are scaled quadratically whereas the pacing delays for the latter case are scaled linearly. The objective is to prevent congestion by metering data packets into the network slower than they can traverse it. This source-throttling technique is called quadratic backoff.

We derive a mathematical model for pacing-delay computations in a packet-radio network that demonstrates quadratic backoff introduces unfair stable equilibrium points. In other words, quadratic backoff causes a persistent situation to occur in which some routes get more bandwidth that others.

Quadratic backoff was considered for SURAP as a congestion-avoidance mechanism, but was eliminiated due to performance problems. Recent versions of SURAP employ the linear-backoff pacing algorithm at the head of a route as well as along a route.

2. Simulation Results

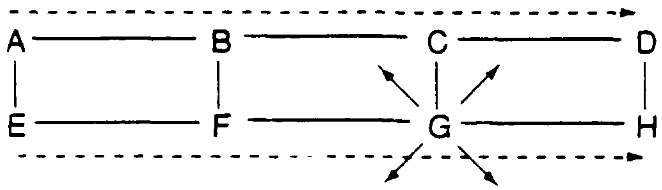

Figure 2.1: Sample Network

Our model of pacing was motivated by the simulation of the network of Figure 2.1. In this network:

1. A sends to D at maximum rate.
2. E send to H at maximum rate.
3. G crashes halfway through the simulation.

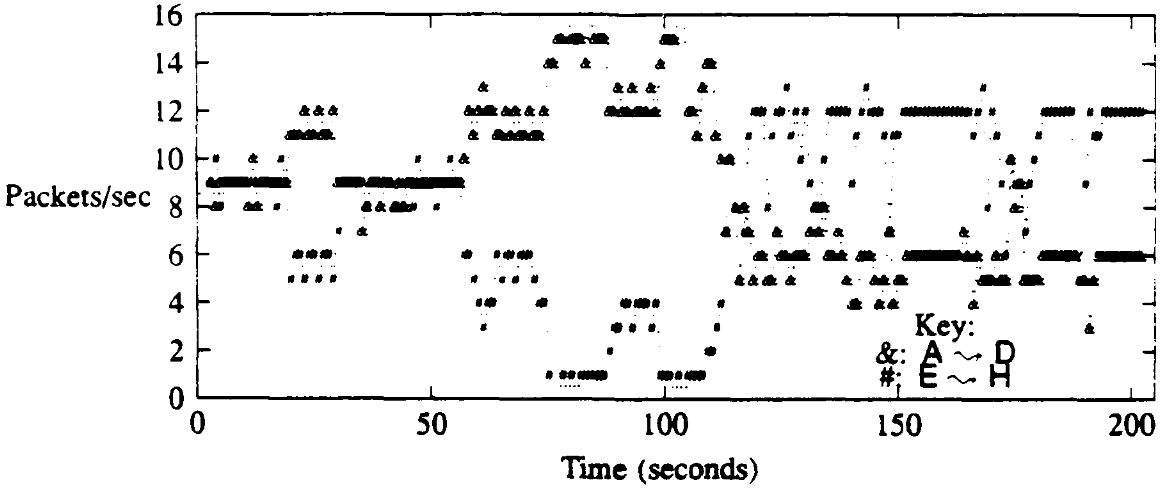

Figure 2.2: Throughput

Figure 2.2 compares the throughputs of the two paths after G crashes. Initially, A ⇝ B's traffic share switches between nearly-exact integer multiples of the E ⇝ H's share. Furthermore, each integer-multiple share-allocation is maintained for several seconds with very few fluctuations. This happens until 120 seconds have elapsed; then, the E ⇝ H path stabilizes at a traffic share that is twice the share of the A ⇝ B path.

For this network, once G crashes there are two routing alternatives for the E ⇝ H path: E may route either through A or through F. These choices are illustrated in figure 2: the dotted line illustrates the path through A while the dashed line illustrates the path through E. On closer examination of the simulation we observed that E routes through A for the first 120 seconds and that E routes through F afterwards.

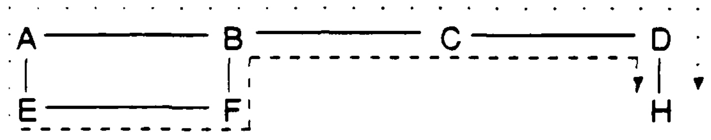

Figure 2.3: Route alternatives

We now investigate these two cases:

2.1 E routes through F

Here, A and F both forward packets to B. Because B is the bottleneck, the throughput of each path is determined by node A's and node F's pacing value to node B. F will get a much higher traffic share because its pacing is equal to 3D (linear backoff) where D is a smoothed average of the time for B to forward packets. However, A's pacing is equal to $M \times D$ (quadratic backoff) where [2]:

$$M = 3 + \frac{13}{213} D \tag{2.1}$$

Fairness queueing helps alleviate this situation. Fairness queueing [1], is a FIFO queuing discipline, with additional stipulations that give queueing priority to infrequent senders; this prevents nodes from hogging the resources of a neighbor. With fairness queueing, B gives transmission preference to packets from A. But this will not eliminate the unfairness; our simulations give the E ⇝ H path twice the traffic share of the A ⇝ D path.

2.2 E routes through A

In this case, the A ⇝ D path usually maintains a much higher traffic share than the E ⇝ H path. The traffic share for the A ⇝ D path is based on the pacing from A to B, which, in turn, is based on B's forwarding delay. The forwarding delay is very small because B does nothing but forward A's packets. However, the traffic share for the E → H path is much smaller because it is based on the delay from E to A. This is larger than the delay from A to B because A forwards its own packets as well as E's packets.

3. Model

We now develop a model of pacing when E routes through A. We use the model to show that the A $\rightsquigarrow$ D traffic share has several equilibrium values that are integer multiple of the traffic share of the E $\rightsquigarrow$ H traffic share.

3.1 Sample Unfair Allocations

Before developing the pacing model, we first give a few examples of E $\rightsquigarrow$ H pacing versus A $\rightsquigarrow$ D pacing. These examples illustrate situations where the A $\rightsquigarrow$ D path can maintain a traffic share that is much larger than the E $\rightsquigarrow$ H path.

To evaluate the pacing delays, we first evaluate forwarding delays. In our simulation the smallest possible measured forwarding delay is 20 msec. This is determined by the packet sizes and the simulated processing times. This is almost always the forwarding delay that A measures to B, since B forwards only node A's packets.

Therefore, given:

$$\begin{aligned} Pacing &= \begin{cases} 3D + \frac{13}{213}D^2 & \text{for source traffic} \\ 3D & \text{for intermediate traffic} \end{cases} \qquad (3.1) \\ D &= 20 \text{ msec} \end{aligned}$$

We obtain:

$$Pacing(\text{A} \rightsquigarrow \text{D}) \text{ at A} = 84.4 \text{ msec} \qquad (3.2)$$

$$Pacing(\text{E} \rightsquigarrow \text{H}) \text{ at A} = 60 \text{ msec} \qquad (3.3)$$

The forwarding delay that E measures to A is the sum of two values:

FWD: The portion of the forwarding time spent processing and transmitting the packet. Therefore, FWD is the smallest possible forwarding delay that E can measure to A: 20 msec.

W: The amount of time that the packet waits at A to be transmitted due to pacing. Note that W cannot exceed the E $\rightsquigarrow$ H pacing delay at A of 60 msec because fairness queueing insures that A will forward E's packet before sending any additional packets originating at A.

We now consider the pacing delay at E for the E $\rightsquigarrow$ H path. The following table illustrates some sample pacing delays for the E $\rightsquigarrow$ H path for some values of W:

W	*Pacing*
0 msec	84.4 msec
30 msec	302 msec
60 msec	630 msec

These delays show that the throughput on the E $\rightsquigarrow$ H path can be considerably less than the throughput on the A $\rightsquigarrow$ D path. For example, when W is 30 msec, the A $\rightsquigarrow$ D path has about three times the throughput as the E $\rightsquigarrow$ H path.

Nonetheless, the variable W is not uniformly distributed between 0 and 60 msec. The next section presents a dynamic model where we solve for equilibrium values of W.

3.2 Equilibrium

After some time, the network dynamics may reach equilibrium. If this occurs, we are interested in evaluating the possible equlilibrium solutions. Before describing these solutions, we first give definitions.

We say that the pacing for a link is in equilibrium if:

$$W_{i+k} = W_i \text{ for some integer k and for i sufficiently large} \qquad (3.4)$$

where W_i is the waiting time at A for the *ith* packet from E.

This definition implies the waiting time will repeat with a period k. When k = 1, we say the system is at an *equilibrium point*; otherwise, we say the system is in an *equilibrium orbit* with period k.

A special case of equilibrium is an equilibrium point or orbit that cannot be reached under any initial conditions other than the equilibrium point or orbit. We call this condition *unstable equilibrium* .

We now address four questions concerning equilibrium:

1. Does the system always reach equilibrium?
2. What are the equilibrium values?
3. Are there any unstable equilibrium values?
4. What are the most probable equilibrium values?

3.3 Difference-Equation Modeling

To answer these questions, we model our system as a set of difference equations. Appendix A gives the derivation of these equations. The equations are:

$$D_i = 20 + W_i \tag{3.5}$$

$$FD_{i+1} = \begin{cases} \frac{3}{4}FD_i + \frac{1}{4}D_i & \text{if } D_i > FD_i \\ \frac{7}{8}FD_i + \frac{1}{8}D_i & \text{otherwise} \end{cases} \tag{3.6}$$

$$P_i = \frac{13}{213}D_i^2 + 3D_i \tag{3.7}$$

$$W_{i+1} = \begin{cases} W_i + 144.4 - P_i & \text{if } P_i < W_i + 84.4 \\ 84.4K_i + 60 + W_i - P_i & \text{if } P_i < W_i + 84.4K_i + 60 \\ 0 & \text{otherwise} \end{cases} \tag{3.8}$$

$$K_i = \lceil (P_i - W_i)/84.4 \rceil \tag{3.9}$$

Where:

D_i is the amount of time for A to forward the ith E $\leadsto$ H packet.

FD_i is the smoothed average of all D_j for $j < i$

P_i is the intertransmission time at caused by pacing between packets i and i + 1

W_i is the amount of time that the ith E $\leadsto$ H packet waits to be sent at A

K_i is the instantaneous ratio of the throughput on A $\leadsto$ D to the throughput on E $\leadsto$ H

3.4 Theoretical Equilibrium Points

Recall that an equilibrium point occurs when $W_i = W_{i+1}$. Therefore, there an equilibrium point occurs when 60 + 84.4K = pacing. The following table displays all of the possible equilibtrium points which we obtain by solving for W_i given a value for K.

K	*Wait time at* A	*Pacing at E*
1	9.92	144.4
2	21.40	228.8
3	31.16	313.2
4	39.79	397.6
5	47.63	482.0
6	54.84	566.4

Note that K cannot exceed six because W has a maximum value of 60 msec by above equations. Therefore, the above table is exhaustive. This analysis shows that it is possible to reach an equilibrium point where one path has six times the network share of another path.

3.5 Difference-Equation Computations

Although the above table gives the only equilibrium points, we do not know if any are reachable. The system may never reach equilibrium, or it may reach an equilibrium orbit.

We would like to examine the set of difference equations to determine if the system stabilizes. However, this is too difficult given their complexity. Instead, we chose to evaluate this system of difference equations using a Monte-Carlo simulation and observe the response. We evaluated the system with several runs, using several sets of initial values for W_0 and D_0.

We made the following observations:

1. The system always reached an equilibtrium point. The equilibrium points depended on the values for W_0 and D_0.
2. The most common equilibrium point was the one that assigns one path three times the share of the other.
3. When we set D_0 to the initial value assigned by SURAP, both paths usually stabilized to equal traffic shares.
4. The equilibrium point at K = 4 is unstable. All other equilibrium points are stable.

We also changed the algorithm so that the smoothing functions never varied. That is,

$$FD_{i+1} = \frac{3}{4} FD_i + \frac{1}{4} D_i \text{ for all cases} \tag{3.10}$$

This is a slight variation of equation 3.6. We changed the algorithm to see how generalizable the results are. Specifically, we wondered if an equilibrium point could always be reached, and if there were special properties to the K = 4 case to make things unstable

This model also gave interesting results:

1. We observed stability for all K.
2. We observed an equilibrium orbit. Specifically, we observed:

$$P_{i+2} = P_i \tag{3.11}$$

The model shows that the network should reach equilibrium and that it is common for the network to stabilize to a point where one path has three times the traffic share of the other. Our SURAP3 simulations demonstrate this as well.

4. Summary and Conclusions

This example shows how quadratic backoff can give unfair traffic shares. Our simulations show that this network will typically stabilize at a point where one path has three times the traffic share of the other. Furthermore, it is theoretically possible to stabilize where one path has six times the share of the other.

Our steady-state analysis can be generalized to other similar topologies. Specifically, this behavior will occur in topologies with routes whose second node is the end-to-end source of another route.

Linear backoff eliminates these problems. Quadratic backoff has been replaced by linear backoff in SURAP.

This example is interesting because it provides insight into the dynamics of SURAP. Specifically, it demonstrates instances of synchronization that may continually and systematically assign certain paths a very small traffic share. This makes it difficult for some nodes to communicate and therefore impacts the network's survivability.

Appendix A. Difference-Equation Derivations

To derive the set of difference equations for the system, we evaluate W_{i+1}, given W_i and P_i. Here, P_i is the pacing delay between packets i and $i+1$ sent from E to A.

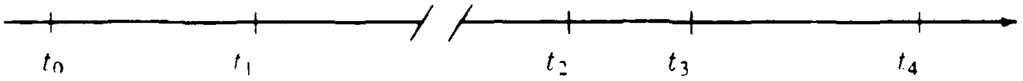

Figure A.1: Time Line

Figure A describes events that occur at A: These are:

t_0: Packet i from E arrives at A from E.

t_1: A forwards packet number i.

t_2: A forwards the last A ⇝ D packet.

t_3: Packet number i + 1 arrives at A from E.

t_4: A forwards packet i + 1.

We now describe equations that relate t_1 through t_4.
First:

$$t_1 = t_0 + W_i, \tag{A.1}$$

by the definition of W_i.

Let K_i be the number of A ⇝ H packets that A forwards between forwarding packet i and packet i + 1. In other words, K_i is the ratio of the throughput of the A ⇝ D path to the throughput of the E ⇝ H path. (Note that fairness queueing insures that $K_i > 0$.)

This means:

$$t_2 = t_1 + K_i P(\text{A} \rightarrow \text{B for A} \rightsquigarrow \text{D packets}) \tag{A.2}$$

Which, by substituting equation 3.2, reduces to:

$$t2 = t_1 + 84.4K_i \tag{A.3}$$

Note that:

$$t_3 = t_0 + P_i \tag{A.4}$$

by the definition of pacing and that:

$$t_4 = t_3 + W_{i+1} \qquad \text{(A.5)}$$

by the definition of W.

We now wish to evaluate W_{i+1}. To do so, we consider three cases:

Case 1: Packet i+1 arrives before A sends a single A $\rightsquigarrow$ D packet.

Case 2: Packet i+1 arrives before its pacing timer at A expires.

Case 3: Packet i+1 arrives after its pacing timer at A expires.

Case 1: Packet i+1 arrives before A sends a single A $\rightsquigarrow$ D packet. For this case, fairness queueing makes packet i + 1 wait for A to send exactly one A $\rightsquigarrow$ D packet.

The conditions for this case mean that:

$$t_3 < t_1 + P(\text{A} \rightarrow \text{B for A} \rightsquigarrow \text{D packets}) = t_1 + 84.4. \qquad \text{(A.6)}$$

Combining this with equations A.1, A.4, and A.5 gives:

$$P_i < W_i + 84.4 \qquad \text{(A.7)}$$

The results for this case mean that:

$$\begin{aligned} t_4 &= t_1 + P(\text{A} \rightarrow \text{B for A} \rightsquigarrow \text{D packets}) \\ &\quad + P(\text{A} \rightarrow \text{B for E} \rightsquigarrow \text{H packets}) && \text{(A.8)} \\ &= t_1 + 144.4 && \text{(A.9)} \end{aligned}$$

Combining this with equations A.1, A.4, and A.5 gives:

$$W_{i+1} = W_i + 144.4 - P_i \qquad \text{(A.10)}$$

Case 2: Packet i+1 arrives before its pacing timer at A expires The conditions of this case mean that:

$$t_3 < \text{pacing expiration time} \qquad \text{(A.11)}$$

where

$$\text{pacing expiration time} = t_2 + P(\text{A} \rightarrow \text{B for E} \rightsquigarrow \text{H packets}) = t_2 + 60 \qquad \text{(A.12)}$$

Combining this with equations A.1, A.4, and A.5 gives:

$$P_i < W_i + 84.4K_i + 60 \qquad \text{(A.13)}$$

Case 3: Packet i+1 arrives after its pacing timer at A expires For this case,

$$t_4 = \text{pacing expiration time} = t_2 + 60 \qquad \text{(A.14)}$$

And therefore,

$$W_{i+1} = 84.4K_i + 60 + W_i - P_i \qquad \text{(A.15)}$$

Recall that K_i is the number of A $\rightsquigarrow$ D packets that A sends between forwarding packets i and i + 1. Therefore, K_i is the largest integer such that:

$$t_1 + K_i P(\text{A} \rightarrow \text{B for A} \rightsquigarrow \text{D packets}) < t_3 \qquad \text{(A.16)}$$

Substituting equations A.1, A.4, and 3.2 gives:

$$K_i = \lfloor (P_i - W_i) / 84.4 \rfloor \qquad \text{(A.17)}$$

The conditions of this case mean:

$$t_3 > \text{pacing expiration time} = t_2 + 60 \qquad \text{(A.18)}$$

Combining this with equations A.1, A.4, and A.5 gives:

$$P_i > W_i + 84.4K_i + 60 \qquad \text{(A.19)}$$

Once the pacing timer has expired, packet i + 1 is forwarded as soon as it arrives at A, and so:

$$W_{i+1} = 0 \qquad \text{(A.20)}$$

Bibliography

[1] John Jubin. Current packet-radio network protocols. In *IEEE INFOCOM*, March 1985.

[2] *Suran Protocol (SURAP) Version 1.3 Working Specification*. Technical Report SRNDOC 3. Defense Advanced Research Prohects Agenct, April 1989.